Social Network Archetypes and Vertex Similarity, Graph Matching and the Generalized Eigenvalue Problem

Charalampos E. Tsourakakis
Department of Mathematical Sciences, CMU

14 October 2011

Abstract

Vertex similarity and graph matching are two significant problems with numerous important applications in different scientific fields, such as data mining, social network analysis, computer vision, biology, chemistry and many more. A key problem towards the design of successful algorithms is finding a good definition of similarity.

In this work we provide novel perspectives on both problems. Inspired by the concept of archetypal analysis introduced by Cutler and Breiman [24], we propose for the vertex similarity problem a geometric optimization problem which allows us to express each vertex uniquely as a convex combination of few extreme types of vertices. We use this mapping of vertices to vectors of mixture coefficients to define vertex similarity. Furthermore, our method has the advantage of supporting efficiently several types of queries such as "which other vertices are most similar to this vertex?" by the use of the appropriate data structures. With respect to the graph matching problem between two graphs G, H, we propose the generalized condition number $\kappa(L_G, L_H)$ of the Laplacian matrix representations of G, H –a quantity widely used in numerical analysis– as a measure of graph similarity. We show that this objective has a solid theoretical basis and propose a deterministic and a randomized graph alignment algorithm.

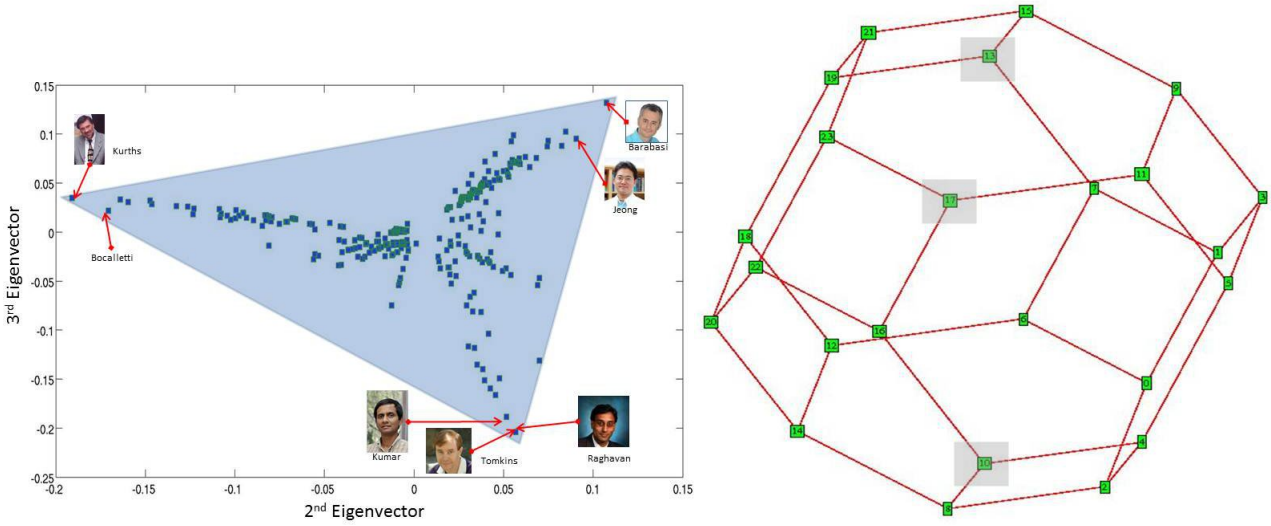
We validate our algorithms on both synthetic and real data. We present promising and interesting findings in our experimental results. Furthermore, we discuss in detail several aspects of our work and propose several research directions.

1 Introduction

Similarity is a significant concept perceived since ancient times, see, e.g., the Nicomachean Ethics by Aristotle [5]. It has been an object of study of various scientific fields, ranging from psychology and philosophy to chemistry and computer science. It is a major philosophical problem to define similarity, if any widely acceptable definition exists. Even in the context of this work which is graph and vertex similarity, a major step towards a successful quantification of similarity between two vertices of a graph or two graphs is finding a good definition. Typically upon having a definition, one can come up with several algorithms and heuristics to find similar vertices within a graph or high-quality graph matchings between two graphs. The applications of graph similarity have a broad range and are of central role in numerous fields. Indicatively we report applications in computer vision (e.g., optical character recognition, biometric identification), chemistry (e.g., chemical compunds comparison [38]), databases (e.g., query optimization [79], structure matching [60]), biology (e.g., protein-protein interaction networks, phylogenetics [39]), information retrieval (e.g., automatic extraction of synonyms in a monolingual dictionary [16]), social networks (e.g., link prediction [57], viral marketing [52]), World Wide Web (e.g., anomaly detection [63]). Furthermore as "birds of a feather flock together", understanding whether or not two vertices (e.g., two users) of a network (e.g., online social network) are similar will automatically imply an improved ability to predict the emergence of an edge between them (e.g., online friendship). Notice that this improved prediction ability may be used for malicious purposes as well, e.g., privacy attacks [36].

In this work we are interested in two problems, both closely related to the concept of similarity. On purpose, we state the problems abstractly, using quotes in several places, in order to emphasize that a major contribution of our work are two novel formalizations of these questions. The first problem is: *given an undirected graph $G(V, E)$ and two vertices $u, v \in V$, how "similar" are u and v?* The second problem one is: *given two graphs $G(V, E_G)$, $H(V, E_H)$, is there a permutation of the vertices of H that "reveals any similarities" between G and H? Can we find such a permutation?* We shall call from now on the first and the second problem as the vertex similarity and the graph matching problem respectively.

We propose two novel approaches to the aforementioned problems. Our approach to the first problem is geometric. Specifically, in order to quantify the similarity of two vertices we find an informative embedding of the graph in the Euclidean space and learn a simplex with minimum possi-



(a) Archetypes and Vertex Similarity

(b) Graph Matching

Figure 1: Read Section 1 for the accompanyting details. (a) Minimum area 2-simplex $\mathcal{S}$ for an informative embedding of the largest component of the Netscience network G, see Table 1 for the dataset details. $\mathcal{S}$ allows us to express each data point as a unique convex combination of its extreme points and hence call two vertices of G similar if their corresponding mixture coefficients are close. (b)Permutahedron $\mathcal{P}$ of the symmetric group S_4. The 3 shaded vertices of $\mathcal{P}$ define a hypothetic the set of isomorphisms between G and H.

ble volume which encloses the data points. Fig. 1(a) shows a minimum area 2-simplex $\mathcal{S}$ for a normalized Laplacian graph embedding [13] of the largest component of the Netscience network, see Table 1. This allows us to express each data point and hence each vertex of the Netscience graph, as a *unique* convex combination of the simplex vertices. Comparing the mixture coefficients of two vertices allows us to quantify their similarity. For instance, using the Euclidean distance between the mixture coefficients (the smaller the distance is, the more similar the vertices are) we find that the pair 'Ravi Kumar' and 'Andrew Tomkins' is highly similar. Furthermore, the 3 extreme points of $\mathcal{S}$ correspond to three different archetypes, i.e., types of research. Specifically, the three vertices of the simplex lie close to Kurths and Bocalletti, Barabasi and Jeong, Kumar, Raghavan, Tomkins and Rajagopalan which are respectively three authoritative groups of researchers on social networks. Our method has the important advantage of supporting *efficiently* queries of the type "which other vertices are most similar to this vertex?", "which are the most dissimilar vertices to this vertex?" etc. by the use of the appropriate data structures, e.g., see [7].

Our approach to the graph matching problem introduces a novel criterion of similarity between two graphs G, H, the *generalized condition number* of their Laplacian matrix representations. Its choice has a solid theoretical basis. Specifically, consider Figure 1(b) which shows the permutahedron $\mathcal{P}$ of the symmetric group S_4 with three shaded vertices which correspond to three hypothetical permutations which make graph H identical to graph G. Theorem 4.1 proves that our proposed randomized algorithm in the limit of $\lambda \to +\infty$ ($\lambda \geq 1$ is a parameter in our algorithm) converges to the uniform distribution over the shaded set, i.e., $\pi\Big(\text{Shaded Vertex}\Big) = 1/3$, $\pi\Big(\text{Non-Shaded Vertex}\Big) = 0$.

The paper is organized as follows: Section 2 presents briefly related work to vertex similarity and graph matching and the theoretical preliminaries for our work. Sections 3 and 4 present a novel approach to the vertex similarity and the graph matching problem respectively. Section 5 shows an experimental validation and evaluation of our proposed methods. Section 6 discusses several aspects of our work and proposes new research directions. Finally, Section 7 concludes the paper.

2 Related Work

In this Section we present related research work and theoretical preliminaries. Due to the large amount of reseach on the vertex similarity and the graph matching problems which additionally spans a wide range of scientific fields it is impossible to exhaust the literature here. The interested reader is urged to read the bibliographies of the cited papers herein.

2.1 Vertex Similarity According to [49] the problem of vertex similarity had received less attention compared to other problems such as community detection, vertex centralities etc. until 2005. The same claim holds in the author's opinion more or less today and most of the existing research work is dominated by the same idea: two vertices are simi-

lar if their neighbors are similar. To no surprise, the recursive nature of this idea leads to recursive algorithms. Before describing the widely used recursive algorithms it's worth pointing out that other measures of similarity exist: the number of common neighbors, Jaccard's coefficient, Salton's coefficient, the Adamic/Acar coefficient [3] etc. These measures have significant shortcomings. For instance, two vertices may be highly similar even if they share no common neighbors. For a fairly detailed discussion see [49].

Probably, the algorithm that has influenced and motivated a large and significant part on vertex similarity is Kleinberg's HITS algorithm [44]. Interestingly, Blondel et al. [15, 16] generalized HITS and provided a general scheme for finding the similarity of two vertices. Jeh and Widom proposed the Simrank algorithm [41] to compute all-pairs vertex similarities in a graph. Since then many improvements and extensions of the SimRank algorithm have appeared e.g., [37, 55]. Leicht et al. propose another recursive measure of similarity closely resembling the centrality measure of Katz [49]. Not surprisingly again, as in HITS algorithm [44] many such algorithms both in their formulation and in the proof of convergence are closely connected to spectral graph theory. Spectral graph theory and specifically the theory of random walks provides the basis for a rich set of similarity measures including, e.g., commute times [26, 28] and the closely related methods including graph kernel methods, e.g., [28, 78]. More recursive schemes have been proposed over the last years, including [39, 83]. Needless to say that finding similar vertices has plenty of applications. Indicatively, we report link recommendation [57], schema matching [60] and privacy attacks [36].

Our geometric perspective on the problem of vertex similarity in Section 3 has not been considered in the literature to the best of our knowledge.

2.2 Archetypal Analysis The idea of archetypal analysis was born by Breiman during his work on predicting the next-day ozone levels [24]: each day should be expressed as a mixture of "extreme" or "archetypal" days. Culter and Breiman introduced archetypal analysis and proposed an alternating minimization procedure [25]. Archetypal analysis has important practical applications in various fields including dynamical system analysis [73], computational biology [40], marketing [64] and many more.

2.3 Spectral Unmixing Spectral unmixing is a central problem in spectral imaging. Specifically, according to [42] "..spectral unmixing is the procedure by which the measured spectrum of a mixed pixel is decomposed into a collection of constituent spectra, or endmembers, and a set of corresponding fractions, or abundances." Keshava [42] surveys existing algorithms for this problem. Of special interest to us is the geometric approach, inspired by Craig's seminal work [23]. Several algorithms have been proposed for this problem, [27, 56, 75, 76] and have found applications in several domains including computational biology [74].

The computational complexity of the algorithms is a function of the dimensionality k of the simplex. Specifically, when $k = 2$ there exist efficient algorithms for finding the minimum area enclosing triangle [61]. When $k = 3$ Zhou and Suri give an algorithm with complexity $O(n^4)$ [84]. Packer showed that the problem is NP-hard when $k \geq \log(n)$ [62].

2.4 Graph Matching A large amount of research work has been devoted to the graph matching problem by several research groups, see, e.g., [2]. Umeyama's influential paper began a whole line of research [77]. Specifically, Umeyama proposed that instead of trying to find a permutation matrix P that minimizes $||PA_GP^T - A_H||$ where A_G, A_H are the adjacency matrix representations of graphs G, H one may relax the problem to finding an orthogonal matrix P' that minimizes the same objective. Then one can find a closed form solution for the optimal P'. Specifically, many methods relax the constraint of searching for a permutation matrix P, i.e., one of the vertices of the Birkoff polytope [18] (it is a polytope whose vertices correspond to permutation matrices, see the closely related permutahedron in Figure 1(b)), to finding doubly stochastic matrices. For the case of graph similarity between graphs with different number of vertices typically a bipartite graph is first created which is widely known as the compatibility graph. The graph matching problem can be formulated as an integer quadratic problem (IQP) which can be tackled in various ways. Dominating approaches include semidefinite programming [65], spectral approaches [11, 19, 45, 58, 69], linear programming relaxations [4, 43] and the popular graduated assignment method [31]. The latter, relaxes the IQP into a non-convex quadratic program and solves a sequence of convex optimization approximation problems. Blondel et al. [15, 16] use a generalization of HITS method [44] to find graph matchings. As we mentioned in Section 2.1 their method is also applicable to the vertex similarity problem. Other approaches include the combination of the Singular Value Decomposition and the Expectation-Maximization algorithm [53, 54], Belief Propagation [12] and kernel-based methods [70, 80].

2.5 Generalized Condition Number The fundamental problem of linear algebra is solving the linear system of equations $Ax = b$ [33]. The convergence rate of several iterative linear solvers including the popular conjugate gradient method [8] depends on the distribution of the eigenvalues of the coefficient matrix A. In the special case of A being a symmetric positive semidenite matrix the convergence rate is a function of the ratio of the largest to the smallest non-zero eigenvalue, also known as the condition number of matrix

A. In the case of a preconditioned linear system the corresponding quantity that determines the rate of convergence of the solver, e.g., preconditioned conjugate gradient [8], is the generalized condition number. The definition follows:

DEFINITION 1. (GENERALIZED CONDITION NUMBER [32]) *Let A, B be two real matrices with the same null space $\mathbb{K}$. λ is a generalized eigenvalue of the ordered pair of matrices (A, B), also called pencil, if there exists a vector $x \notin \mathbb{K}$ such that $Ax = \lambda Bx$. Let $\Lambda(A, B)$ the set of generalized eigenvalues of the pencil (A, B). The generalized condition number $\kappa(A, B)$ is defined as the ratio of the maximum value $\lambda_{\max}(A, B)$ to the minimum value $\lambda_{\min}(A, B)$.*

For every unit norm vector x the following double inequality holds:

$$\lambda_{\min}(A, B)x^T Bx \leq x^T Ax \leq \lambda_{\max}(A, B)x^T Bx. \tag{2.1}$$

For the special case of interest where the pencil (L_G, L_H) is a pair of Laplacian matrices of two connected graphs G, H on the same vertex set the condition number is given by the following expression:

$$\kappa(L_G, L_H) = \left(\max_{x^T 1=0} \frac{x^T L_G x}{x^T L_H x}\right)\left(\max_{x^T 1=0} \frac{x^T L_H x}{x^T L_G x}\right).$$

Notice that we since G, H are connected their null space is the same and specifically the span of the all ones vector $\mathbf{1}$ [22]. This also explains why we require vector x to be orthogonal to $\mathbf{1}$ in the above expression.

Generalized eigenvalue problems of a special form have several important applications in computer science. The seminal paper of Shi and Malik [68] considered the generalized eigenvalue problem $Lx = \lambda Dx$ where L is the Laplacian matrix representation of the graph and D is a diagonal matrix whose i-th entry is the degree of the i-th vertex to produce image segmentations. Exactly the same problem was used to produce structure-informative graph embeddings by Belkin and Niyogi [13]. Studying the spectrum of the pencil (L, D) is equivalent to studying the so called normalized Laplacian matrix representation [22]. The form of this problem appears a lot in structural mechanics where L is typically called the stiffness matrix and D the mass matrix. It's worth pointing out –to the best of our knowledge– that we were not able to find any other applications of generalized eigenvalue problems in the literature where D is not diagonal.

3 VertexSim: Vertex Similarity via Simplex Fitting

Before we delve into the details of the VertexSim algorithm, we wish to outline its central idea: along the lines of archetypal analysis [24], there exist few "extreme" types of vertices. Let us call these vertices pure. The vast majority of vertices however is a mixture of those pure types of vertices. We formalize the notion of "extreme" in a geometric sense. Specifically, we take advantage of the geometry of the network (for a discussion whether an assumption of the existence of geometric structure holds or not, see Section 6) by embedding it to a low-dimensional Euclidean space $\mathbb{R}^k$. However, instead of taking the convex hull of the points as the set of pure points we fit a simplex, i.e., the convex hull of $k + 1$ affinely independent points, to the cloud of points. The vertices of the simplex are the pure types. The rationale is that we can express uniquely each other point as a convex combination of the pure points and hence make a quantitative analysis of vertex similarity. Furthermore, having for each vertex in the graph a vector of mixture coefficients, we can build data structures, e.g., [7] that answer several types of queries such as "which are the three vertices most similar to vertex v?" etc. Among all simplexes that fit the cloud of points we favor the one with the smallest volume, inspired by the seminal work of Craig [23]. We shall call the vertices of the fitted simplex *social network archetypes* since they correspond to the pure types of vertices.

There are several methods to obtain an informative embedding of the data points in the Euclidean space. The majority of them are spectral [48]. Recently, more expensive but structure preserving methods have been proposed and are based on semidefinite programming [67]. In our experiments we choose the k smallest non trivial eigenvectors, i.e., the eigenvectors corresponding the k smallest, non-zero eigenvalues, of the normalized Laplacian [13]. There exist a wide variety of off-the-shelf algorithms that find a minimum volume enclosing simplex [27, 56, 74, 75, 76] and good implementations publicly available. We use the algorithm developed by the author and his co-authors in a previous paper [74] which solves the following optimization problem, where $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{R}^k$ is the cloud of points, $K = [v_0|\ldots|v_k]$ is a simplex in $\mathbb{R}^k$ and $\theta_i \in [0, 1]^{k+1}$ for $i = 1, \ldots, n$ is the vector of mixture coefficients of point i:

$$\begin{aligned} \min_{K} \quad & : \quad \sum_{i=1}^{s} |x_i - K\theta_i|_p + \gamma \log \texttt{vol}(K) \\ \forall \theta_i \quad & : \quad \theta_i^T \mathbf{1} = 1,\ \theta_i \succeq 0 \end{aligned} \tag{3.2}$$

For completeness reasons we include the steps of our method in Algorithm 2. In the second step, we use the Nedler-Mead method [47] to optimize Program 3.2. In the third and fourth step we compute the similarity between every possible pair of vertices and insert the mixture coefficient vectors as points to a data structure that supports the type of queries we are interested in, e.g., nearest neighbor queries.

4 CondSim: Graph Similarity and the Generalized Condition Number

As we have already outlined from Section 1, the graph matching problems poses a challenging question: *What is a good measure of similarity?* Typically, upon having a good

Algorithm 1 VertexSim

Require: Connected, undirected graph $G([n], E)$. Dimension k. Parameter γ.

(1) Embed the graph using the k smallest non trivial eigenvectors of the normalized Laplacian of G.

(2) $[K, \{\theta_i\}_{i\in[n]}] \leftarrow$ Solve Optimization Problem 3.2 using the Nelder-Mead method [47].

(3) For every pair of vertices (i, j) compute Pearson's correlation coefficient ρ_{ij} between the mixture coefficient vectors θ_i, θ_j and set the similarity value $\text{Sim}(i, j)$ equal to ρ_{ij}.

(4) Add points $\{\theta_i\}_{i\in[n]}$ to a data structure supporting nearest neighbor search queries.

objective one can come up with various algorithms to optimize it. In this section we provide a novel measure of graph similarity between two graphs G, H. Our measure is the *generalized condition number* between the Laplacian matrix representations L_G, L_H of the two graphs. Recall from Section 2.5 that the generalized condition number plays an important role in Numerical Analysis. In Section 4.1 we provide our main theoretical result with our proposed algorithms, namely CondSimMC and CondSimGradDescent. In Section 4.2 we provide further insight on the proposed measure of similarity between graphs.

4.1 Theoretical Result and Algorithms Let $G([n], E_G), H([n], E_H)$ be connected graphs on n vertices (named for simplicity $\{1, 2, .., n\} = [n]$) and L_G, L_H their Laplacian matrix representation respectively. Also, let $\Lambda(L_G, L_H)$ and $\kappa(L_G, L_H)$ be the set of generalized eigenvalues and the generalized condition number of the pencil (L_G, L_H) [32, 33]. We use S_n to denote the symmetric group, i.e., the group whose elements are all the permutations of the set $[n]$ and whose group operation is the composition of such permutations. We denote with $L_G^{(\sigma)}$ where $\sigma \in S_n$ the Laplacian matrix representation of the graph G whose vertex set has been renamed according to σ, i.e., $v \mapsto \sigma(v)$ for all $v \in [n]$. Our main result is the next theorem.

THEOREM 4.1. *Let Ω be the state space represeting the set of all permutations $\{\sigma : \sigma \in S_n\}$, and $f : \Omega \to \mathbb{R}^+$ be a function defined by $f(\omega) = \kappa(L_G, L_{H^{(\omega)}})$ for all $\omega \in \Omega$. Also, fix $\lambda \geq 1$ and define $\pi_\lambda(\omega) = \frac{\lambda^{-f(x)}}{Z(\lambda)}$ where $Z(\lambda) = \sum_{x\in\Omega} \lambda^{-f(x)}$ is the normalizing constant that makes π_λ a probability measure. We define a Metropolis chain where we allow transitions between two states if and only if they differ by a transposition as follows: if $f(\omega_1) < f(\omega_2)$ the Metropolis Chain accepts the transition $\omega_1 \to \omega_2$ with probability $\lambda^{f(\omega_1)-f(\omega_2)}$ otherwise always accept it.*

As $\lambda \to +\infty$ the stationary distribution π_λ of the Metropolis chain converges to the uniform distribution over the global minima of f. Furthermore, if $G \sim H$, i.e., G, H are isomorphic, then π_λ converges to the uniform distribution over the set of isomorphisms $\{\sigma : L_G = L_{H^{(\sigma)}}, \sigma \in S_n\}$.

Before we prove Theorem 4.1, consider two graphs G, H, on 4 vertices each. Figure 1(b) shows the permutahedron with 4!=24 vertices, each corresponding to one of the 24 possible permutations of the set $[4] = \{1, 2, 3, 4\}$ [18]. Assume that there exist three permutations $\sigma_1, \sigma_2, \sigma_3$ such that $L_G = L_{H^{(\sigma_i)}}$ for $i = 1, 2, 3$. The Metropolis chain [50] we define in Theorem 4.1 has the property to converge in the limit of $\lambda \to +\infty$ to the uniform distribution over the shaded set, i.e., $\pi\Big(\text{Shaded Vertex}\Big) = 1/3$, $\pi\Big(\text{Non-Shaded Vertex}\Big) = 0$. When G and H are not isomorphic there exists no state in Ω such that $\kappa(L_G, L_{H^{(\omega)}})$=1 and hence for any $\sigma \in S_n$ $\kappa(L_G, L_{H^{(\omega)}}) > 1$. By standard properties of the Metropolis chain we know that the chain converges to the minima of the generalized condition number. In Section 4.2 we explain further why this is a good property using the theory of support tree preconditioners. It's also worth pointing out that by using transpositions, we can reach any state/permutation of Ω since every permutation is a product of transpositions.

Algorithm 2 CondSimMC

Require: L_G, L_H the Laplacian matrix representation of G, H respectively. MAXITER (Maximum number of iterations). $\lambda \geq 1$. TOLERANCE. { RAND() generates a number uniformly at random in $[0, 1]$} {σ initialized to the identity permutation}

$\sigma \leftarrow (1, 2, .., n)$
$i \leftarrow 0$
while $i \leq$ MAXITER **do**
  $i \leftarrow i + 1$
  $\sigma_{\text{neigh}} \leftarrow$ Tranpose two elements of σ (uniformly at random).
  if $\kappa(L_G, L_{H^{(\sigma_{\text{neigh}})}}) - \kappa(L_G, L_{H^{(\sigma)}}) \leq 0$ **then**
    $\sigma \leftarrow \sigma_{\text{neigh}}$
    CN$\leftarrow \kappa(L_G, L_{H^{(\sigma)}})$
  else
    if RAND() $\leq \lambda^{\kappa(L_G, L_{H^{(\sigma)}}) - \kappa(L_G, L_{H^{(\sigma_{\text{neigh}})}})}$ **then**
      $\sigma \leftarrow \sigma_{\text{neigh}}$
      CN$\leftarrow \kappa(L_G, L_{H^{(\sigma)}})$
    end if
  end if
  if CN $\leq 1+$TOLERANCE **then**
    BREAK
  end if
end while
return (σ, CN)

Proof. Recall that the Laplacian representation of a con-

nected graph is a symmetric, positive semidefinite matrix and that the dimension of the null space is 1 (the all-ones vector $\mathbf{1}$) [22]. Consider now the generalized eigenvalue problem $L_G x = \lambda L_H x$. The pencil (L_G, L_H) is Hermitian semidefinite. Therefore there exists a basis of generalized eigenvectors [33]. Notice that the all-ones vector $\mathbf{1}$ is a generalized eigenvector with corresponding generalized eigenvalue 0, see also [82] pp.35-36. Let $\Lambda(L_G, L_H) = \{0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{n-1}$ be the set of generalized eigenvalues. Then, $\kappa(L_G, L_H) = \frac{\lambda_{n-1}}{\lambda_1}$[1]. We prove that $\kappa(L_G, L_H) = 1$ if and only $G \sim H$.

• $\kappa(L_G, L_H) = 1 \Rightarrow G \sim H$:

The generalized eigenvalues are $\lambda(L_G, L_H) = (0 = \lambda_0 < 1 = \lambda_1 = \ldots = \lambda_{n-1})$. Let $(\mathbf{1} = u_0, u_1, \ldots, u_{n-1})$ be the corresponding generalized eigenvectors which *form a basis*. Define $X = L_G - L_H$. Notice that $Xu_i = 0$ for all $i = 0, .., n-1$. Hence, $X = 0$ and therefore $L_G = L_H \rightarrow G \sim H$.

• $G \sim H \Rightarrow \exists \sigma \in S_n$ SUCH THAT $\kappa(L_G, L_{H^{(\sigma)}}) = 1$:

Since $G \sim H$ there exists a permutation $\sigma \in S_n$ such that $L_G = L_{H^{(\sigma)}}$. Simply, by substituting the eigenvectors $\{u_i\}_{i=0,..,n-1}$ of L_G in $L_G x = \lambda L_{H^{(\sigma)}} x = \lambda L_G x$ we obtain that the generalized eigenvalues are $(0, 1, 1, .., 1)$ and the corresponding eigenvectors $(\mathbf{1} = u_0, u_1, \ldots, u_{n-1})$. Hence, $\kappa(L_G, L_{H^{(\sigma)}}) = 1$.

Now, define $\Omega^* = \{\omega \in \Omega : f(\omega) = f^* = \min_{x \in \Omega} f(x)\}$. Since our chain is a Metropolis chain [50], its stationary distribution is π_λ. Therefore,

(4.3)

$$\lim_{\lambda \to +\infty} \pi_\lambda(\omega) = \lim_{\lambda \to +\infty} \frac{\lambda^{f(\omega)}/\lambda^{f^*}}{|\Omega^*| + \sum_{\omega \in \Omega - \Omega^*} \lambda^{f(\omega)}/\lambda^{f^*}} = \frac{I(\omega \in \Omega^*)}{|\Omega^*|}$$

where $I(\alpha \in A)$ is an indicator variable equal to 1 if element α belongs to set A, otherwise 0. If G, H are isomorphic then $f^* = 1$ and therefore the above result suggests that the Metropolis Chain converges to the uniform distribution over the set $\{\sigma : L_G = L_{H^{(\sigma)}}, \sigma \in S_n\}$.

Algorithm 2 shows the pseudocode for CondSimMC which was described in Theorem 4.1. The algorithm takes as input the two Laplacians L_G, L_H, the parameter MAXITER which is the maximum number of steps that we allow the Metropolis chain to perform, the parameter $\lambda \geq 1$ and the parameter TOLERANCE≥ 0 which quantifies an early stopping criterion that allows CondSimMC to terminate when the condition number is at most 1+TOLERANCE.

[1]Notice that despite the fact that our matrices are positive semidefinite, and not positive definite, this doesn't cause any real problem with respect to defining the generalized condition number since L_G, L_H have the same null space.

Algorithm 3 shows an alternative way to minimize the generalized condition number which we call CondSimGradDescent. The input of CondSimGradDescent does not require the parameter λ since it is deterministic. Algorithm 3 takes the parameter $\epsilon > 0$ which quantifies the least amount of progress required by the algorithm to keep iterating. This may help in avoiding extremely incremental improvements which don't significantly improve the graph matching but cost a lot computationaly. It's worth pointing out that we set in our experiments $\epsilon = 0$ but non-the-less its existence may be convenient in certain executions. CondSimGradDescent performs a gradient descent with respect to the generalized condition number using transpositions. On the one hand Algorithm 3 tends to be computationally more aggressive in the sense that it always moves to a state/permutation which results in a smaller generalized condition number. On the other hand Algorithm 2 due to the randomization is likely to avoid local minima in contrast to the determininstic Algorithm 3. Both CondSimMC and CondSimGradDescent return the permutation which defines the best graph alignment found and the corresponding condition number.

The complexities of our algorithms clearly depend on the choice of algorithm that solves the generalized eigenvalue problem. Specifically, let $f(L_G, L_H)$ be the corresponding running time as a function of the two Laplacians. Also let q abbreviate the maximum number of iterations MAXITER. Then the total running time is upper bounded by $O(qn^2 f(L_G, L_H))$ since we perform q steps and at each step we compute the generalized condition number for the $\binom{n}{2}$ possible transpositions. In our experiments we use the algorithm of Golub and Ye [32]. The speed of convergence is given in Lemma 1, p. 8 [32]. For our purposes, since we set the number of iterations (which are matrix-vector multiplications) of the Golub-Ye algorithm to a constant we may assume that the running time that computing the smallest non-trivial and the largest generalized eigenvalue of the pencil (L_G, L_H) is linear in the total number of edges $|E_G| + |E_H| = O(m)$ where $m = \max(|E_G|, |E_H|)$. If m is too large, i.e., $m >> n \log n$, one can use the developed theory of spectral sparsifiers to speed up the generalized condition number computations. Specifically, one may perform first the popular Spielman-Srivastava sparsification [71] on both Laplacians L_G, L_H, obtain spectrally equivalent matrices $\tilde{L}_G, \tilde{L}_H$ and apply the CondSimGradDescent on the latter Laplacians.

4.2 Further Insight: Theory of Support Trees We proved in Theorem 4.1 that when the generalized condition number is 1, then indeed G, H can be perfectly matched, i.e., G, H are isomorphic. To complete the justification of our rationale behind the choice of our measure of similarity we need to explain why does a value close to 1 imply a good graph alignment. The answer lies in the theory of

Algorithm 3 CondSimGradDescent

Require: L_G, L_H the Laplacian matrix representation of G, H respectively. MAXITER (Maximum number of iterations). TOLERANCE. $\epsilon > 0$. $\{\sigma$ initialized to the identity permutation$\}$
$\sigma \leftarrow (1, 2, .., n)$
$i \leftarrow 0$
while $i \leq$ MAXITER **do**
$i \leftarrow i + 1$
$\sigma^* \leftarrow \arg\max_{\sigma' \in S'} \kappa(L_G, L_{H^{(\sigma)}}) - \kappa(L_G, L_{H^{(\sigma')}})$
where S' is the set of all permutations which differ from σ. a single transposition.
if $\kappa(L_G, L_{H^{(\sigma)}}) - \kappa(L_G, L_{H^{(\sigma')}}) > \epsilon$ **then**
$\sigma \leftarrow \sigma^*$
CN$\leftarrow \kappa(L_G, L_{H^{(\sigma)}})$
else
BREAK
end if
if CN $\leq$ 1+TOLERANCE **then**
BREAK
end if
end while
return $(\sigma$, CN)

support preconditioners [1]. Preconditioning is one of the "holy grails" of scientific computing. Pravin Vaidya who did not publish any of his work brought remarkable new ideas in this field, giving birth to the field called today combinatorial scientific computing. The following facts come from Gremban's thesis [34] which continued and significantly extended Vaidya's ideas. These complete the justification of our rationale. In the following, let A, B be Laplacian matrices.

DEFINITION 2. (SUPPORT) *The support $\sigma(A, B)$ of matrix B for A is the greatest lower bound over all τ such that $\tau B - A$ is positive semidefinite, i.e., $\sigma(A, B) = \lim inf\{\tau : \tau B - A \succeq 0\}$.*

DEFINITION 3. (CONGESTION & DILATION) *An embedding of H into G is a mapping of vertices of H onto vertices of G, and edges of H onto paths in G. The dilation $d(G, H)$ of the embedding is the length of the longest path in G onto which an edge of H is mapped. The congestion $g_e(G, H)$ of an edge e in G is the number of paths of the embedding that contain e. The congestion $g(G, H)$ of the embedding is the maximum congestion of the edges in G.*

The following facts has been proved in Gremban's Ph.D. thesis [34], see also [35]:

Fact 1: The support number $\sigma(A, B)$ is bounded above by the maximum product of dilation and congestion over all embedding of A into B.

Fact 2: $\kappa(A, B) \leq \sigma(A, B)\sigma(B, A)$, Lemma 4.8 [34].

Fact 2, in combination with Fact 1, shows that the generalized condition number is closely related to the goodness of two embeddings, i.e., of H into G and vice versa. When both the dilation and the congestion of the embeddings –or in our terminology of the alignments– are small then the generalized condition number is small. The reader interested in further details and proofs is urged to read Chapter 4 of Gremban's thesis [34]. In combination with Theorem 4.1 this justifies our choice of the generalized condition number as a measure of graph similarity. For further discussion on other properties of the generalized condition number read Section 6.

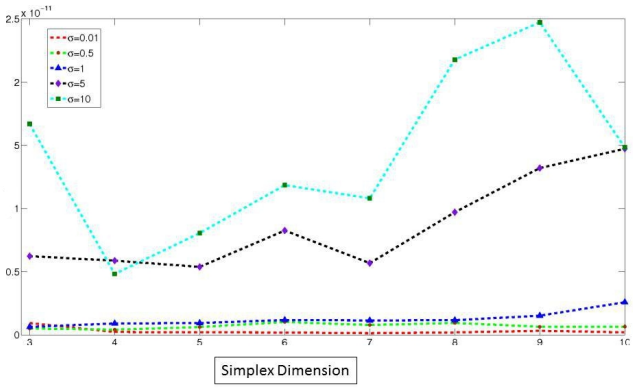


Figure 2: Performance of simplex fitting on 1000 points drawn uniformly at random from a randomly generated k-simplex perturbed by Gaussian noise $N(0, \sigma^2)$. Figure plots the sum of Euclidean distances of the $k + 1$ reconstructed simplex vertices from the $k + 1$ true vertices as a function of the dimensionality k of the simplex for five different standard deviations $\sigma = 0.01, 0.5, 1, 5, 10$. Notice that the simplex fitting method (essentially) perfectly recovers the true simplex in all cases.

5 Experiments

Name (Abbr.)	Nodes (n)	Edges (m)
⊙ Netscience	1589	2742
⊙ Football	115	613
⊙ Political Books	105	441
⋆ Erdös '72	5488	7085
⋆ Erdös '82	5822	7375
⋆ Erdös '02	6927	8472
⋆ Roget Thesaurus	1022	3648

Table 1: Datasets

In this Section we provide an experimental evaluation of the proposed algorithmic schemes, VertexSim and CondSim. The Section is organized as follows: In Sections 5.1, 5.2 we describe the datasets we used in our experiments and the experimental setup. In Sections 5.3, 5.4 we provide an experimental evaluation of our proposed methods respectively.

5.1 Datasets Table 1 summarizes the real-world datasets we used for our experiments. Whenever neccessary, graphs

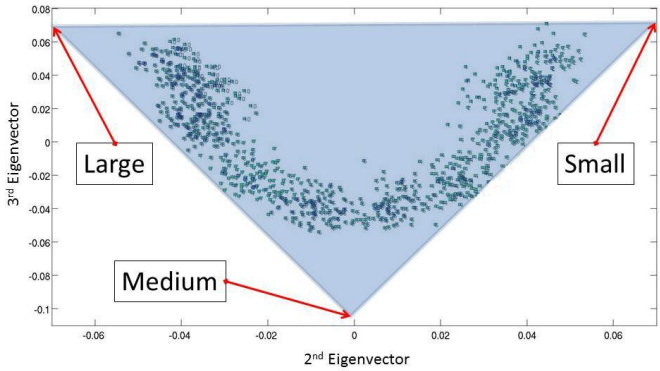


Figure 3: 2-simplex fitted on a random stratified network. VertexSim correctly assigns higher similarity values to vertices of the same age. The three vertices of the fitted 2-simplex conceptually represent the concepts 'senior/large age' (8-10), 'middle-aged/medium age'(4-7) and 'young/small age' (1-3).

are made undirected, unweighted and self loops were removed. Datasets annotated with ⊙ and ⋆ are available from `http://www.cise.ufl.edu/research/sparse/matrices/Newman/index.html` and `http://www.cise.ufl.edu/research/sparse/matrices/Pajek/index.html` respectively. We pick small and medium sized networks on purpose since the geometric structure is striking. See Section 6 for further discussion concerning social networks and geometric structure.

We also generate several synthetic datasets. Specifically, for Section 5.3.1 we generate a cloud of points where each point is chosen uniformly at random from a random k-simplex (see Appendix [50]) and a random stratified social network, see Section IIIA of [49]. Stratified networks model the phenomenon according to which individuals making connections with similar others with respect to some criterion, e.g., income, age. For each vertex we pick an age from 1 to 10, chosen uniformly at random. Two vertices with age i and j respectively are connected with probability $p_0 e^{-\alpha \Delta t}$ where $\Delta t = |i - j|$. The parameters are set to $\alpha = 0.8$ and $p_0 = 0.1$. Finally, for Section 5.4 we generate random graphs of two types. Erdös-Rényi-Gilbert graphs [17] and R-MAT graphs [20]. For the former we use $p = 0.5$ and for the latter the parameters are set to $[a = 0.55, b = 0.1; c = 0.1, d = 0.25]$.

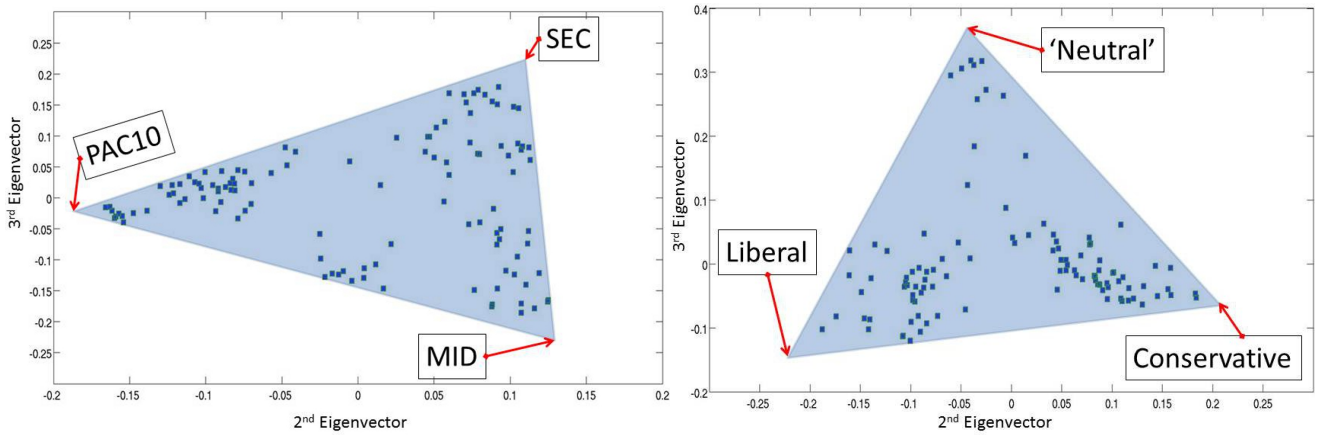
5.2 Experimental Setup and Implementation Details The experiments were performed on a single machine, with Intel Xeon CPU at 2.83 GHz, 6144KB cache size and and 50GB of main memory. Our algorithms are implemented in MATLAB. We used the Golub-Ye algorithm [32] to compute condition numbers. The results we show are obtained for setting the parameter γ was set to 1 and the dimensionality k of the embedding equal to 2. Clearly our method is valuable when k is larger than 3 where visualization is impossible (see also Section 2.3 for a discussion of the computational complexity as a function of k). Here we report results for $k = 2$ for visualization purposes. It's worth pointing out two more facts concerning our experimental section: first VertexSim for the datasets we used was indifferent for the value of parameter γ since there were no outliers in any of the embeddings and secondly we experimented with higher values of k (from 3 to 5) obtaining highly interpretable results. The parameters of CondSimGradDesc were set to MAXITER=200, TOLERANCE=0, $\epsilon = 0$ for all experiments in Section 5.4.

A final remark with respect to the experiments of CondSim in Section 5.4: it is a well known fact that a permutation can be decomposed in cycles and that a random permutation has in expectation $O(\log n)$ cycles [81]. Therefore if in our experiments we generate only permutations chosen uniformly at random we are restricting ourselves with high probability to permutations which have a common structure. To avoid a potential artifact in our experimental results, we generate permutations with a ranging number of cycles. We use a simple recursive algorithm [81] to generate a permutation with k cycles uniformly at random in our experiments, see p.33 [81]. Finally, we use third-party software and specifically the code of [7, 12] and Jeremy Kepner's Rmat code implementation.

5.3 VertexSim at Work Sections 5.3.1 and 5.3.2 present the results of VertexSim on synthetic and real data respectively.

5.3.1 Synthetic Data We validate the VertexSim algorithm in two ways: first we verify that it can successfully recover the simplex $\mathcal{S}$ and hence the mixture coefficients of data points sampled uniformly at random from $\mathcal{S}$ and secondly we evalueate its performance on a stratified network. Figure 2 shows the performance of our fitting method as a function of the simplex dimension (x-axis) for five different standard deviations $\sigma = 0.01, 0.5, 1, 5, 10$ (5 lines) for a randomly generated k-simplex. The quality of the performance (y-axis) is quantified as the sum $\sum_{i=1}^{k+1} ||v_i - \tilde{v}_i||$ where $\tilde{v}_i$ is the reconstructed vertex of the k-simplex. The performance is excellent as Fig. 2 shows. The average running time for four executions is 0.0071 and the variance 1.4×10^{-6}. It's worth pointing out that Tsang's [75] algorithm results also in the exactly same simplex.

Figure 3 shows the performance of VertexSim for a stratified network with $\alpha = 0.8$ and $p_0 = 0.1$ and ages ranging from 1 to 10, picked uniformly at random for every vertex. Specifically Fig. 3 shows the fitted 2-simplex. Upon performing step 3 of Algorithm 1, it becomes apparent that pairs of vertices with the same age are significantly more similar than vertices with different ages, as a good vertex



(a) Football (b) Political books

Figure 4: Minimum area 2-simplexes for the (a) Football network (b) Political books network. In both cases VertexSim provides significant mining capabilities for extracting pairs of highly similar vertices and concepts. For more see Section 5.3.2.

similarity algorithm should have as its output. Furthermore the three vertices of the 2-simplex correspond to the three concepts 'senior/large age', 'middle-aged/medium age' and 'young/small age'.

5.3.2 Real-world Data Figure 4(a) shows the minimum area fitted 2-simplex for the American football college network, whose vertices correspond to teams and edges to games among them. According to [30] the teams are divided into conferences containing around 812 teams each and the frequency of games between members of the same conference is higher than between members of different conferences. The three vertices of the fitted simplex correspond to three conferences PAC 10, SEC and MID. Furthermore, VertexSim using the fitted mixture coefficients assigns higher similarity to vertices of the same conference. Similar remarks hold for Figure 4(b) which shows the minimum area fitted 2-simplex for the political books network whose vertices represent books and edges copurchasing by the same buyer. The three vertices of the simplex correspond to liberal, convervative and 'neutral' books. The latter is in quotes since around that vertex of the simplex there also exist few scattered liberal and conservative data points/books. The data points whose mixture coefficient corresponding to the 'neutral' vertex is close to 1 should probably be characterized as moderate rather than neutral. For both datasets, the vectors of mixture coefficients provide us a way to determine vertex similarities in an interpretable way. Unfortunately due to space constraints we can only include the figures shown in Figure 4. In an extensive version of this work, more figures shall be added which also highlight with heatmaps the block structure of the above two datasets. Concerning other datasets on which we have applied our method (including the Roget Thesaurus network, Erdös collaboration networks in 1972, 1982 and 2002) the results of VertexSim are excellent overall. Indicately we report few highly similar pairs of vertices according to VertexSim: (musician, poetry), (melody, poetry), (voice, hearing) from the Roget Thesaurus network and networks that (Vojtech Rödl, Noga Alon), (Joel Spencer, Janòs Pach) from the Erdös collaboration network in 1972.

	Erdös-Renyi			R-MAT		
# Vertices	8	16	32	8	16	32
Condsim	6/7	5/15	7/31	5/7	5/15	11/31
Belief Propagation[12]	0/7	0/15	0/31	0/7	0/15	0/31

Table 2: Results of our method versus the Belief Propagation method of [12] on various random networks for permutations whose number of cycles ranges from 1 to # vertices-1. The fractions indicate how many times did an algorithm find a permutation which makes the original graph and its permuted version exactly the same.

5.4 CondSim at Work Sections 5.4.1 and 5.4.2 present the results of CondSim on synthetic and real data respectively. In the following we use the gradient descent version of our algorithm.

5.4.1 Synthetic Data We compare CondSim with the Belief Propagation based method of [12] for few synthetic datasets in the following way. We generate a graph (Erdös-Rényi-Gilbert and RMAT) of n vertices and a permutation with k cycles, where k ranges from 1 to $n-1$. Notice that we don't consider the identity permutation with n cycles. We permute the graph according to the random permutation and see whether the graph matching methods can align perfectly the original graph and its permuted version. We use two types of random graphs, namely Erdös-Rényi-Gilbert and Rmat graphs with 8, 16, 32 vertices. Table 2 shows the

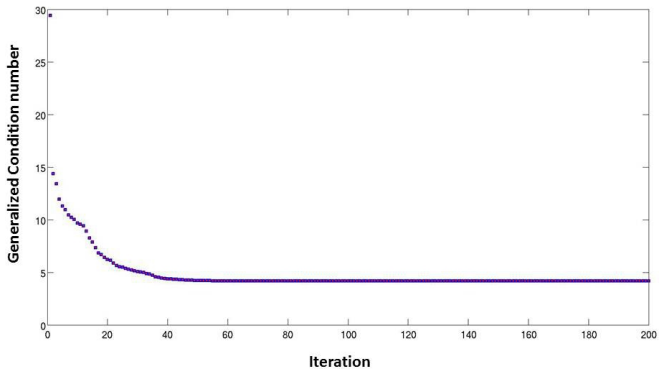


Figure 5: A random execution of CondSimGradDesc on the Football network with a random permutation. The method in the first few steps makes large improvements in the generalized condition number but then it gets stuck to a local minimum of value 4.16.

results. As we see, CondSim outperforms significantly [12]. It's worth pointing out again that this is a validation test and that fast graph isomorphism tests exist, e.g., for Erdös-Renyi graphs see Ch.3 [17]. An interesting trend observed from the Table and verified in other experiments as well, is that the fewer the cycles of the permutation, the easier CondSim gets "stuck" to local minima. On the positive side when the number of cycles is small CondSim finds an optimal alignment fairly easily.

5.4.2 CondSim as a Post-Processing Tool Due to the computational cost of CondSim, a realistic use of it in applications would be as a post processing tool. We show an example of how CondSim can be used as a postprocessing tool to improve the graph alignment in combination with the Belief-Propagation based method of [12]. We perform the following simple experiment: we consider the graph G of the *Football* network, see Table 1. We generate a permutation uniformly at random and permute the labels of G accordingly. We apply the function *netalignbp()* allowing all possible edges in the bipartite graph of [12]. The number of fixed points of the permutation we obtained is 0. The alignment produced by [12] has recognized correctly 50 out of the 115 correct vertex to vertex assignments. The generalized condition number equals 29.4. Applying the CondSimGradDesc method to the alignment obtained from Belief Propagation, we obtain a generalized condition number of value 4.16 resulting in 74 correct assignments. Figure 5 shows the progress of CondSim during this process and illustrates the fact that it gets stuck to a local minimum. Understanding this behavior is an interesting research problem, including several others which we discuss in the following Section.

6 Discussion & Research Directions

The work we presented in Sections 3 and 4 opens several new research directions, certain of which we discuss in this section. We picked the simplex as the geometric object to be fitted. Given the simplex, we "reduce"[2] the problem of finding the similarity of two vertices to comparing their corresponding, unique, convex combination coefficients. This approach has two important advantages, namely supporting interesting types of queries such as 'which other vertices are most similar to this vertex?" and extracting useful knowledge for our network by examining the archetypes. It appears from our experimental results that simplexes capture well the geometry of small and medium sized social networks. Two natural questions come up: Is there always geometric structure is social networks? Can we fit more other geometric objects such as simplicial complexes to capture more complex geometric structure? Leskovec et al. [51] have studied extensively properties of large scale networks and it appears that there exists strong geometric structure in small and medium sized networks like the ones we studied in Section 5 but this structure typically decays as the size of the network grows. The answer to the second question is an interesting research problem, see also [21]. Concerning our optimization problem, several other optimization techniques can be applied due to its special form, see for instance [66]. An experimental study of these methods for our problem is a future task. Finding statistically meaningful ways to pick parameters k, γ is also a task to be completed. In our experiments, empirically small values for k, γ gave excellent results. Another aspect concerns the graph embedding. We used a normalized Laplacian graph embedding [13] in our experiments. There exist new techniques which are computationally more expensive than spectral-based dimensionality reduction methods [48] but guarantee a stronger preservation of geometric, see for instance [67]. Finally, an interesting question concerns the relation of our method to community detection. For instance, what is the result of clustering the vectors of mixture coefficients and how does it compare to existing clustering methods, e.g., [30]?

Concerning our second algorithmic scheme, an interesting problem is to extend it to cases where the two graphs have a different number of vertices, i.e., $|V_G| < |V_H|$. Even if several heuristics can be thought of such as adding several one-degree vertices to the graph G until they become of the same order $|V_H|$ and using perturbation theory [72] to quantify the effect, the approach of Steiner tree preconditioners is more promising, see [46]. An interesting view on our method is the following: it is well known that we can view a graph as an electrical network [26] and that for a given graph G, $x^T L_G x$ equals the total energy consumed by the network where x is the vector of voltages on the endpoints/vertices. Taking into account Inequality 2.1 and the above fact, we are trying to align the two electrical networks in such way that

[2] Notice that it is not a real reduction since there is no uniquely accepted definition of similarity of two vertices.

for any experiment we do, i.e., setting the voltage vector x, the total energies consumed are similar. Setting the voltages in such way to produce cuts [14] it also becomes evident that the CondSim algorithmic scheme tries to make the cut structures of the aligned graphs as similar as possible. We want to outline that our work does not attempt to make any claims neither for the theoretical graph isomorphism problem [6] (which for many classes of graphs is solvable in polynomial time, e.g., [10], and remains open for specific classes of graphs which are highly symmetric, e.g., strongly regular graphs) nor for the practical [59]. The complexity of graph isomorphism remains open despite sophisticated attempts, see [29] and [9]. However, a detailed theoretical analysis of our method is an interesting research problem as is the less ambitious goal of understanding its performance in simple cases such as paths or trees. Our method provides a new similarity objective which has good reasons to exist. Finally, our method can be used in combination with other methods, e.g., Belief Propagation [12], as a post-processing tool in order to improve the alignment. In Section 5 we show such a use and verify in practice the value of our method. Using it in other applications such as shape matching is a direction to be pursued in the near future.

7 Conclusion

In this work we propose two novel algorithmic schemes VertexSim and CondSim for the vertex similarity and the graph matching problem respectively. Our main contributions are two-fold. First, we introduce a geometric perspective of an archetypal analysis on social networks which allows to answer efficiently queries of the type "which other vertices are most similar to this vertex?" using appropriate data structures such as (approximate) nearest neighbor data structures [7]. Secondly, we propose the generalized condition number of two Laplacians as a measure of similarity of graphs based on the theory of support tree preconditioners [34]. We justified in Section 4 in detail why this is a good measure of similarity. We introduce three new algorithms which give interesting and promising results. Our work opens numerous new research directions. We discuss several such directions in detail in Section 6.

Acknowledgements

Research supported by NSF Grant No. CCF-1013110. The author thanks Ioannis Koutis, Gary Miller for several discussions on the theory of support tree preconditioners and Jure Leskovec for discussions concerning the geometry of social networks. Also the author thanks Dan Spielman for pointing out references [29, 9].

References

[1] *Support Theory for Preconditioning*, `http://www.sandia.gov/~bahendr/support.html`.

[2] Pattern Recognition Letters - Special issue: *Graph-based representations in pattern recognition.* Vol. 24(8), May 2003

[3] Adamic, L. Adar, E: *Friends and neighbors on the web.* Social Networks, Vol. 25(3), pp. 211230 (2003)

[4] Almohamad, H. A., Duffuaa, S. O.: *A Linear Programming Approach for the Weighted Graph Matching Problem.* IEEE Trans. Pattern Anal. Mach. Intell., Vol. 15(5), pp. 522-525 (1993)

[5] Aristotle: *Nicomachean Ethics.* (350 B.C.)

[6] Arvind, V., Torán, J.: *Isomorphism Testing: Perspective and Open Problems.* Bulletin of the EATCS, Vol. 86, pp. 66-84 (2005)

[7] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., Wu, A. Y.: *An Optimal Algorithm for Approximate Nearest Neighbor Searching.* Journal of the ACM, Vol. 45, p. 891-923 (1998)

[8] Axelsson, O., Barker, V.A.: *Finite element solution of boundary value problems: theory and computation.* Society for Industrial and Applied Mathematics (2001)

[9] Babai, L.: *The double permutation polytope is NP-hard.* Available at `http://people.cs.uchicago.edu/~laci/polytope.pdf`

[10] Babai, L., Grigoryen, D., Mount, D. M.: *Isomorphism of graphs with bounded eigenvalue multiplicity.* Proceedings of the fourteenth annual ACM symposium on Theory of computing (STOC), pp. 310-324 (1982)

[11] Bai, X., Yu, H., Hancock, E.R: *Graph Matching using Spectral Embedding and Alignment.* ICPR Vol. 3, pp. 398-401 (2004)

[12] Bayati, M., Gerritsesn, M., Gleich, D.F., Saberi, A., Wang, Y.: *Algorithms for large, sparse network alignment problems.* In Proceedings of the 9th IEEE International Conference on Data Mining, pp. 705-710 (2009)

[13] Belkin, M., Niyohi, P.: *Laplacian Eigenmaps for Dimensionality Reduction and Data Representation.* Neural Computation, Vol. 15(6), pp. 1373-1396 (2003)

[14] Benczúr, A., Karger, D.: *Approximating s-t Minimum Cuts in* $\tilde{(}O)n^2$ *Time.* ACM Symposium on Theory of Computing, pp. 47-55 (1996)

[15] Blondel, V. D., Van Dooren, P.: *Similarity matrices for pairs of graphs.* Proc. of the 30th international conference on Automata, languages and programming (ICALP), pp. 739-750 (2003)

[16] Blondel, V. D., Gajardo, A., Heymans, M., Senellart, P., Van Dooren, P.: *A measure of similarity between graph vertices.* SIAM Review, Vol. 46, pp. 647-666 (2003)

[17] Bollobás, B.: *Random Graphs.* Cambridge Studies in advanced mathematics, Second Edition

[18] Burkard, R., Dell'Amico, Martello, S.: *Assignment Problems.* SIAM Monographs on Discrete Mathematics and Applications (2009)

[19] Caelli, T., Kosinov, S.: *An Eigenspace Projection Clustering Method for Inexact Graph Matching.* IEEE Trans. Pattern Anal. Mach. Intell. (PAMI), Vol. 26(4), pp. 515-519 (2004)

[20] Chakrabarti, D., Zhan, Y. Faloutsos, C.: *R-MAT: A recursive model for graph mining.* SIAM Data Mining (2004)

[21] Chazal, F., Guibas, L., Oudot, S., Skraba, P.: *Persistence-based clustering in riemannian manifolds.* Proceedings of the 27th annual ACM symposium on Computational geometry (SOCG), pp. 97-106 (2011)

[22] Chung Graham, F.: *Spectral Graph Theory.* American Mathematical Society (1997)

[23] Craig, M.: *Minimum-volume transforms for remotely sensed data.* IEEE Transactions on Geoscience and Remote Sensing, Vol. 3(3), p. 542-552 (1994)

[24] Cutler, A.: *Remembering Leo Breiman.* Annals of Applied Statistics, Vol. 4(4), pp. 1621-1633 (2010)

[25] Cutler, A., Breiman, L.: *Archetypal Analysis.* Technometrics, Vol. 36, pp. 338-347 (1994)

[26] Doyle, P., Snell. J. L.: *Random Walks and Electric Networks.* Available at `http://arxiv.org/abs/math/0001057`

[27] Ehrlich R, Full W: *Sorting out geology unmixing mixtures.* Use and Abuse of Statistical Methods in the Earth Sciences Oxford University Press, p. 33-46 (1987)

[28] Fouss, F., Pirotte, A., Renders, J-M., Saerens, M.: *Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation.* IEEE Trans. Knowl. Data Eng., Vol. 19(3), pp. 355-369 (2007)

[29] Friedland, S.: *Graph isomorphism is Polynomial.* Available at `http://arxiv.org/PScache/arxiv/pdf/0801/0801.0398v1.pdf`

[30] Girvan, M., Newman, M. E. J.: *Community structure in social and biological networks.* Proc. National Academy of Sciences (PNAS), Vol. 99(12), pp. 7821-78266 (2002)

[31] Gold, S., Rangarajan, A.: *A Graduated Assignment Algorithm for Graph Matching.* IEEE Trans. Pattern Anal. Mach. Intell., Vol. 18(4), pp. 377-388 (1996)

[32] Golub, G., Ye, Q.: *An Inverse Free Preconditioned Krylov Subspace Method for Symmetric Generalized Eigenvalue Problems.* SIAM Journal on Scientific Computing, Vol. 24, pp. 312-334.

[33] Golub, G., Van Loan, C.F.: *Matrix Computations.* JHU Press (1996)

[34] Gremban, K.: *Combinatorial Preconditioners for Sparse, Symmetric, Diagonally Dominant Linear Systems.* Ph.D. Thesis, Carnegie Mellon University (1996)

[35] Guattery, S., Leighton, T., Miller, G.L.: *The path resistance method for bounding λ_2 of a Laplacian.* Combinatorics, Probability and Computing Vol. 8, p. 441-460 (1999)

[36] Hay, M., Miklau, G., Jensen, D., Towsley, D.F., Weis, P.: *Resisting structural re-identification in anonymized social networks.* PVLDB, Vol. 1(1), pp. 102-114 (2008)

[37] He, G., Feng, H., Cuiping, L. Chen, H.: *Parallel SimRank computation on large graphs with iterative aggregation.* Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 543-552 (2010)

[38] Hattori, M., Okuno, Y., Goto, S., and Kanehisa, M.: *Development of a chemical structure comparison method for integrated analysis of chemical and genomic information in the metabolic pathways.* Journal of American Chemistry Society, Vol. 125, pp. 11853-11865 (2003)

[39] Heymans, M., Singh, A. K.: *Deriving phylogenetic trees from the similarity analysis of metabolic pathways.* ISMB (Supplement of Bioinformatics), pp. 138-146 (2003)

[40] Huggins, P., Pachter, L., Strumfels, B.: *Toward the Human Genotope.* Bulletin of Mathematical Biology, Vol. 69(8), pp. 2723-2725 (2007)

[41] Jeh, G., Widom, J.: *SimRank: a measure of structural-context similarity.* Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 538-543 (2002)

[42] Keshava, N.: A Survey of Spectral Unmixing Algorithms. Lincoln Laboratory Journal, Vol. 14(1), pp. 55-78 (2003)

[43] Klau, G.: *A new graph-based method for pairwise global network alignment.* BMC Bioinformatics, 10(Suppl 1):S59 (2009).

[44] Kleinberg, J.: *Authoritative sources in a hyperlinked environment.* Journal of the ACM, Vol. 46, pp. 614-632 (1999)

[45] Knossow, D., Sharma, A., Mateus, D., Horaud, R.: *Inexact Matching of Large and Sparse Graphs Using Laplacian Eigenvectors.* Graph-Based Representations in Pattern Recognition (GbRPR), pp. 144-153 (2009)

[46] Koutis, I.: *Combinatorial and algebraic tools for multigrid algorithms.* Ph.D. Thesis, Carnegie Mellon University, CMU CS Tech Report CMU-CS-07-131 (2007)

[47] Lagarias, J., Reeds, J., Wright, M.H., Wright, P.: Convergence properties of the Nedler-Meader simplex method in low dimensions. SIAM J. of Optimization, Vol. 9(1), pp. 112-147 (1994)

[48] Lee, J.A., Verleysen, M: *Nonlinear Dimensionality Reduction.* Information Science and Statistics

[49] Leicht, E. A., Holme, P., Newman, M. E. J.: *Vertex similarity in networks.* Phys. Rev. E 73, 026120 (2006)

[50] Levin, D., Peres, Y., Wilmer, E.: *Markov Chains and Mixing Times.* American Mathematical Society (2008)

[51] Leskovec, J., Lang, K., Dasgupta, A., Mahoney, M.: *Statistical properties of community structure in large social and information networks.* Proceedings of the 17th International Conference on World Wide Web (WWW), pp. 695-704 (2008)

[52] Leskovec, J., Adamic, L., Huberman, B.A.: *The dynamics of viral marketing.* TWEB, Vol. 1(1) (2007)

[53] Luo, B., Hancock, E.R.: *Structural Graph Matching Using the EM Algorithm and Singular Value Decomposition.* IEEE Trans. Pattern Anal. Mach. Intell., Vol. 23(10), pp. 1120-1136 (2001)

[54] Luo, B., Hancock, E.R.: *Alignment and Correspondence Using Singular Value Decomposition.* SSPR/SPR, pp. 226-235 (2000)

[55] Li, P., Liu, H., Xu Yu, J., He, J. Du, X.: *Fast Single-Pair SimRank Computation.* Proc. of the SIAM International Conference on Data Mining (SDM), pp. 571-582 (2010)

[56] Li, J., Bioucas-Dias, J.: *Minimum Volume Simplex Analysis: A fast Algorithm to Unmix Hyperspectral Data.* IEEE International Geoscience and Remote sensing Symposium IGARSS (2008)

[57] Liben-Nowell, D., Kleinberg, J.: *The Link Prediction Problem for Social Networks.* Proc. of the 12th International Conference on Information and Knowledge Management (CIKM), (2003)

[58] Mateus, D., Horaud, R. P., Knossow, D., Cuzzolin, F., Boyer, E.: *Articulated Shape Matching Using Laplacian Eigenfunctions and Unsupervised Point Registration.* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2008)

[59] McKay, B.: *Practical Graph Isomorphism.* Congressus Numerantium, Vol. 30, pp. 45-87 (1981)

[60] Melnik, S., Garcia-Molina, H., Rahm, E.: *Similarity flooding: a versatile graph matching algorithm.* Proceedings of Eighteenth International Conference on Data Engineering (ICDE), (2002)

[61] O'Rourke, J, Aggarwal, A., Maddila, S., Baldwin, M.: *An optimal algorithm for finding minimal enclosing triangles.* J. Algorithms, Vol. 7(2), pp. 258-269 (1986)

[62] Packer, A.: *NP-hardness of largest contained and smallest containing simplices for V - and H-polytopes.* Discrete Comput. Geom, Vol. 28, pp. 349-377 (2002)

[63] Papadimitriou, P., Dasdan, A., Garcia-Molina, H.: *Web graph similarity for anomaly detection.* Journal of Internet Services and Applications, Vol. 1(1), pp. 19-30 (2010)

[64] Riedesel, P.: *Archetypal Analysis in Marketing Research: A New Way of Understanding Consumer Heterogeneity.* Action Marketing Research, available at `http://www.action-research.com/archtype.pdf`

[65] Schellewald, C., Schnörr, C.: *Probabilistic subgraph matching based on convex relaxation.* Energy Minimization Methods in Computer vision and Pattern Recognition (2005)

[66] Schmidt, M, Fung, G., Rosales, R.: *Fast Optimization Methods for L1 Regularization: A Comparative Study and Two New Approaches.* European Conference on Machine Learning (ECML), (2007)

[67] Shaw, B., Jebara, T.: *Structure preserving embedding.* Proceedings of the 26th Annual International Conference on Machine Learning (ICML), Vol. 382 (2009)

[68] Shi, J., Malik, J.: *Normalized cuts and image segmentation.* IEEE Pattern Analysis and Machine Intelligence, Vol. 22(8), pp. 888 - 905 (2000)

[69] Singh, R., Xu, J., Berger B.: *Pairwise global alignment of protein interaction networks by matching neighborhood topology.* In Proceedings of the 11th Annual International Conference on Research in Computational Molecular Biology (RECOMB), pp. 1631 (2007)

[70] Smalter, A., Huan, J., Lushington, G.: *CPM: a graph pattern matching kernel with diffusion for accurate graph classification.* Technical Report, University of Kansas, ITTC-FY2009-TR-45910-01 (2008)

[71] Spielman, D., Srivastava, N.: *Graph sparsification by effective resistances.* Proceedings of the 40th annual ACM symposium on Theory of computing (STOC), pp. 563-568 (2008)

[72] Stewart, G.W.: *Gershgorin Theory for the Generalized Eigenvalue Problem* $Ax = \lambda Bx$. Math. Comp (1975)

[73] Stone, E., Cutler, A.: *Archetypal analysis of spatio-temporal dynamics.* Physica D: Nonlinear Phenomena Volume 90(3), pp. 209-224 (1996)

[74] Tolliver, D., Tsourakakis, C.E., Subramanian, A., Shackney, S., Schwartz, R.: *Robust unmixing of tumor states in array comparative genomic hybridization data.* Bioinformatics ISMB, Vol. 26(12), pp. 106-114 (2010)

[75] Tsung-Han Chan, Chong-Yung Chi, Yu-Min Huang, Wing-Kin Ma: *Convex analysis based minimum-volume enclosing simplex algorithm for hyperspectral unmixing.* ICASSP, pp. 1089-1092 (2009)

[76] Tsung-Han Chan, Chong-Yung Chi, Yu-Min Huang, Wing-Kin Ma: *A convex analysis-based minimum-volume enclosing simplex algorithm for hyperspectral unmixing.* IEEE Transactions on Signal Processing, Vol. 57(11), pp. 4418-4432 (2009)

[77] Umeyama, S.: *An eigendecomposition approach to weighted graph matchings problems.* Transactions of pattern analysis and machine intelligence, Vol. 10(5) (1998)

[78] Vacic, V., Iakoucheva, L. Lonardi, S., Radivojac, P. *Graphlet kernels for prediction of functional residues in protein structures.* Journal of Computational Biology, Vol. 17(1), pp. 55-72 (2010)

[79] Yan, X, Yu, P. S., Han, J.: *Substructure similarity search in graph databases.* Proceedings of the 2005 ACM SIGMOD international conference on Management of data (SIGMOD), pp. 766-777 (2005)

[80] Wang, H., Hancock, E.R.: *Correspondence matching using kernel principal components analysis and label consistency constraints.* Pattern Recognition, Vol. 39(6), pp. 1012-1025 (2006)

[81] Wilf, H.S.: *East Side, West Side...* Notes, Available at `http://www.math.upenn.edu/~wilf/eastwest.pdf` (1999)

[82] Wisniewski, J.: *On solving the large sparse generalized eigenvalue problem.* Ph.D. Thesis, Univ. of Illinois at Urbana-Champaign, UIUCDCS-R-81-1056 (1981)

[83] Zhao, P. Han, J., Sun, Y.: *P-Rank: a comprehensive structural similarity measure over information networks.* Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM), pp. 553-562 (2009)

[84] Zhou, Y. and Suri, S.: *Algorithms for minimum volume enclosing simplex in* $\mathbb{R}^3$. Proceedings of the Eleventh Annual ACM SIAM Symposium on Discrete Algorithms (SODA), pp. 500509 (2000)